C 8

```
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDDDDDDDDDDD     RRRRRRRRRRRR        IIIIIIIII     VVV         VVV  EEEEEEEEEEEEEEE  RRRRRRRRRRRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRR         RRR        III        VVV         VVV  EEE              RRR         RRR
DDD         DDD  RRRRRRRRRRRR           III        VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD         DDD  RRRRRRRRRRRR           III        VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD         DDD  RRRRRRRRRRRR           III        VVV         VVV  EEEEEEEEEEEE     RRRRRRRRRRRR
DDD         DDD  RRR   RRR              III        VVV         VVV  EEE              RRR   RRR
DDD         DDD  RRR   RRR              III        VVV         VVV  EEE              RRR   RRR
DDD         DDD  RRR   RRR              III        VVV         VVV  EEE              RRR   RRR
DDD         DDD  RRR      RRR           III           VVV   VVV     EEE              RRR      RRR
DDD         DDD  RRR      RRR           III           VVV   VVV     EEE              RRR      RRR
DDD         DDD  RRR      RRR           III           VVV   VVV     EEE              RRR      RRR
DDDDDDDDDDDD     RRR         RRR     IIIIIIIII           VVV        EEEEEEEEEEEEEEE  RRR         RRR
DDDDDDDDDDDD     RRR         RRR     IIIIIIIII           VVV        EEEEEEEEEEEEEEE  RRR         RRR
DDDDDDDDDDDD     RRR         RRR     IIIIIIIII           VVV        EEEEEEEEEEEEEEE  RRR         RRR
```

```
**FILE**ID**LPDRIVER                                J 7
```

```
LL          PPPPPPPP    DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
LL          PPPPPPPP    DDDDDDDD    RRRRRRRR      IIIIII    VV      VV  EEEEEEEEEE  RRRRRRRR
LL          PP      PP  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
LL          PP      PP  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
LL          PP      PP  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
LL          PP      PP  DD      DD  RR      RR      II      VV      VV  EE          RR      RR
LL          PPPPPPPP    DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
LL          PPPPPPPP    DD      DD  RRRRRRRR        II      VV      VV  EEEEEEEE    RRRRRRRR
LL          PP          DD      DD  RR  RR          II      VV      VV  EE          RR  RR
LL          PP          DD      DD  RR  RR          II      VV      VV  EE          RR  RR
LL          PP          DD      DD  RR    RR        II        VV  VV    EE          RR    RR        ....
LL          PP          DD      DD  RR    RR        II        VV  VV    EE          RR    RR        ....
LLLLLLLLLL  PP          DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR      ....
LLLLLLLLLL  PP          DDDDDDDD    RR      RR    IIIIII        VV      EEEEEEEEEE  RR      RR      ....

LL            IIIIII      SSSSSSSS
LL            IIIIII      SSSSSSSS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II      SS
LL              II        SSSSSS
LL              II        SSSSSS
LL              II              SS
LL              II              SS
LL              II              SS
LL              II              SS
LLLLLLLLLL    IIIIII    SSSSSSSS
LLLLLLLLLL    IIIIII    SSSSSSSS
```

Table of contents

```
0000          .TITLE  LPDRIVER - LP11/LS11/LV11 LINE PRINTER DRIVER
0000          .IDENT  'V04-000'
0000
0000    ;
0000    ;****************************************************************************
0000    ;*                                                                          *
0000    ;*  COPYRIGHT (c) 1978, 1980, 1982, 1984 BY                                 *
0000    ;*  DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.                  *
0000    ;*  ALL RIGHTS RESERVED.                                                    *
0000    ;*                                                                          *
0000    ;*  THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED   *
0000    ;*  ONLY IN  ACCORDANCE WITH  THE  TERMS  OF  SUCH  LICENSE  AND WITH THE   *
0000    ;*  INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR  ANY  OTHER   *
0000    ;*  COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY   *
0000    ;*  OTHER PERSON.  NO TITLE TO AND OWNERSHIP OF  THE  SOFTWARE IS  HEREBY   *
0000    ;*  TRANSFERRED.                                                            *
0000    ;*                                                                          *
0000    ;*  THE INFORMATION IN THIS SOFTWARE IS  SUBJECT TO CHANGE WITHOUT NOTICE   *
0000    ;*  AND  SHOULD  NOT  BE  CONSTRUED AS  A COMMITMENT BY DIGITAL EQUIPMENT   *
0000    ;*  CORPORATION.                                                            *
0000    ;*                                                                          *
0000    ;*  DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE  OR  RELIABILITY OF ITS   *
0000    ;*  SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.                 *
0000    ;*                                                                          *
0000    ;*                                                                          *
0000    ;****************************************************************************
0000
0000    ; ABSTRACT:
0000    ;
0000    ;       LP11/LS11/LV11 LINE PRINTER DRIVER
0000    ;
0000    ; AUTHOR:
0000    ;
0000    ;       R. HEINEN 6-SEP-76
0000    ;
0000    ; MODIFIED BY:
0000    ;
0000    ;       V03-011 EMD0085         Ellen M. Dusseault      30-Apr-1984
0000    ;               Add DEV$M_NNM characteristic to DEVCHAR2 so that these
0000    ;               devices will have the "node2" prefix.
0000    ;
0000    ;       V03-010 EMD0084         Ellen M. Dusseault      19-Apr-1984
0000    ;               Fix problem with lowercase p not appearing.  It was
0000    ;               accidently put in the control table so remove it.
0000    ;
0000    ;       V03-009 EAD0150         Elliott A. Drayton      13-Apr-1984
0000    ;               Change the sense of the TRUNCATE branch
0000    ;               and make truncate the default.
0000    ;
0000    ;       V03-008 EAD0147         Elliott A. Drayton      12-Apr-1984
0000    ;               Added support for TAB and TRUNCATE.
0000    ;
0000    ;       V03-007 EMD0077         Ellen M. Dusseault      10-Apr-1984
0000    ;               Modify to make code more efficient.
0000    ;
0000    ;       V03-006 EMD0047         Ellen M. Dusseault      22-Jan-1984
0000    ;               Add new feature, fallback.  The ability to convert 8-bit
```

```
0000   ;                 ascii characters to their 7-bit equivalent representation
0000   ;
0000   ;         V03-005 TCM0001         Trudy C. Matthews        14-Dec-1983
0000   ;                 Change NOP wait loops to use calibrated EXE$GL_UBDELAY cell.
0000   ;
0000   ;         V03-004 EAD0069         Elliott A. Drayton        6-Jan-1983
0000   ;                 Changed default number of lines per page to 66.
0000   ;
0000   ;         V03-003 EAD0068         Elliott A. Drayton       21-Sep-1982
0000   ;                 Correct UCB$L_LP_OFLCNT storage allocation from byte
0000   ;                 to longword. Also reposition code for LP ready test.
0000   ;
0000   ;         V03-002 EAD0067         Elliott A. Drayton       01-Jul-1982
0000   ;                 Change branch instructions for horizontal position tests.
0000   ;
0000   ;         V03-001 KDM0002         Kathleen D. Morse        28-Jun-1982
0000   ;                 Added $DYNDEF, $DCDEF, and $PRDEF.
0000   ;
0000   ;--
```

```
                 0000            .SBTTL  Declarations
                 0000
                 0000    ;
                 0000    ; MACRO LIBRARY CALLS
                 0000    ;
                 0000
                 0000            $CRBDEF                         ;DEFINE CRB OFFSETS
                 0000            $DCDEF                          ;DEFINE DEVICE TYPES
                 0000            $DDBDEF                         ;DEFINE DDB OFFSETS
                 0000            $DPTDEF                         ;DEFINE DPT OFFSETS
                 0000            $DYNDEF                         ;DEFINE DYNAMIC DATA STRUCTURE TYPES
                 0000            $IDBDEF                         ;DEFINE IDB OFFSETS
                 0000            $IODEF                          ;DEFINE I/O FUNCTION CODES
                 0000            $IRPDEF                         ;DEFINE IRP OFFSETS
                 0000            $JIBDEF                         ;DEFINE JIB OFFSETS
                 0000            $LPDEF                          ;DEFINE LINE PRINTER CHARACTERISTICS
                 0000            $MSGDEF                         ;DEFINE SYSTEM MESSAGE TYPES
                 0000            $PCBDEF                         ;DEFINE PCB OFFSETS
                 0000            $PRDEF                          ;DEFINE PROCESSOR REGISTER NUMBERS
                 0000            $SSDEF                          ;DEFINE SYSTEM STATUS CODES
                 0000            $UCBDEF                         ;DEFINE UCB OFFSETS
                 0000            $VECDEF                         ;DEFINE VEC OFFSETS
                 0000
                 0000    ;
                 0000    ; LOCAL SYMBOLS
                 0000    ;
                 0000    ; ARGUMENT LIST OFFSET DEFINITIONS
                 0000    ;
                 0000
        00000000 0000    P1=0                                    ;First function dependent parameter
        00000004 0000    P2=4                                    ;Second function dependent parameter
        00000008 0000    P3=8                                    ;Third function dependent parameter
        0000000C 0000    P4=12                                   ;Fourth function dependend parameter
        00000010 0000    P5=16                                   ;Fifth function dependent parameter
        00000014 0000    P6=20                                   ;Sixth function dependent parameter
        00000780 0000    LP_HRCNT=1920                           ;Timeout value for one hour
                 0000
                 0000    ;
                 0000    ; CHARACTER CODE DEFINITIONS
                 0000    ;
                 0000
        0000000D 0000    C_CR=13                                 ;Carriage return
        0000000C 0000    C_FF=12                                 ;Form feed
        0000000B 0000    C_VT=11                                 ;Verticle tab
        0000000A 0000    C_LF=10                                 ;Line feed
        00000009 0000    C_TAB=9                                 ;Tabulation
                 0000
                 0000    ;
                 0000    ; FLAG REGISTER BIT DEFINITIONS
                 0000    ;
                 0000
        00000001 0000    M_CRPEND=1                              ;Carriage return pending
        00000000 0000    V_CRPEND=0                              ;
                 0000
                 0000    ;
                 0000    ; LP11/LS11/LV11 DEVICE REGISTER OFFSET DEFINITIONS
                 0000    ;
```

```
                0000
                0000                   $DEFINI LP
                0000
                0000    $DEF    LP_CSR             .BLKW   1       ;CONTROL STATUS REGISTER
                0002            $VIELD  LP_CSR,6,<-
                0002                    <IE,,M>,-
                0002                    <DONE,,M>,-
                0002                    >
                0002    $DEF    LP_DBR             .BLKW   1       ;DATA BUFFER REGISTER
                0004
                0004                   $DEFEND LP
                0000
                0000    ;
                0000    ; DEFINE DEVICE DEPENDENT UNIT CONTROL BLOCK OFFSETS
                0000    ;
                0000
                0000                   $DEFINI UCB
                0000
      00000090  0000    .=UCB$K_LENGTH                              ;
                0090
                0090    $DEF    UCB$L_LP_MUTEX   .BLKL   1       ;Line printer UCB mutex
                0094    $DEF    UCB$L_LP_TIMEOUT .BLKL   1       ;Printer problem message timer
                0098    $DEF    UCB$L_LP_OFLCNT  .BLKL   1       ;Offline time counter
                009C    $DEF    UCB$B_LP_CURSOR  .BLKB   1       ;Current horizontal position
                009D    $DEF    UCB$B_LP_LINCNT  .BLKB   1       ;Current line count on page
                009E    $DEF    UCB$B_SPARE      .BLKB   2       ;SPARE UNUSED BYTES
                00A0
      000000A0  00A0    UCB$K_SIZE=.
                00A0                   $DEFEND UCB
```

```
0000          .SBTTL  Driver prologue table and driver dispatch table
0000
0000    ;
0000    ; LOCAL DATA
0000    ;
0000    ; DRIVER PROLOGUE TABLE
0000    ;
0000
0000          DPTAB   -                         ;DEFINE DRIVER PROLOGUE TABLE
0000                  END=LP_END,-              ;End of driver
0000                  ADAPTER=UBA,-             ;Adapter type
0000                  UCBSIZE=UCB$K_SIZE,-      ;UCB size
0000                  NAME=LPDRIVER             ;Driver name
0038          DPT_STORE INIT                    ;Control block init values
0038          DPT_STORE UCB,UCB$B_FIPL,B,8      ;Fork IPL
003C          DPT_STORE UCB,UCB$L_DEVCHAR,L,-   ;Device characteristics
003C                  <DEV$M_REC-               ; Record oriented
003C                  !DEV$M_AVL-               ; Available
003C                  !DEV$M_CCL-               ; Carriage control device
003C                  !DEV$M_ODV>               ; Output device
0043          DPT_STORE UCB,UCB$L_DEVCHAR2,L,-; Device characteristics
0043                  <DEV$M_NNM>               ; prefix name with "node$"
004A          DPT_STORE UCB,UCB$B_DEVCLASS,B,DC$_LP ;Device class
004E          DPT_STORE UCB,UCB$B_DEVTYPE,B,LP$_LP11 ;Device type
0052          DPT_STORE UCB,UCB$W_DEVBUFSIZ,W,132 ;Default buffer size
0057          DPT_STORE UCB,UCB$L_DEVDEPEND,L,-
0057                  <66@24+LP$M_MECHFORM!LP$M_TRUNCATE> ;Printer parameters
005E          DPT_STORE UCB,UCB$B_DIPL,B,20     ;Device IPL
0062          DPT_STORE UCB,UCB$L_LP_MUTEX,W,-1 ;Initialize mutex
0067          DPT_STORE REINIT                  ;Control block re-init values
0067          DPT_STORE CRB,CRB$L_INTD+4,D,LP$INT ;Interrupt service routine address
006C          DPT_STORE CRB,CRB$L_INTD+VEC$L_INITIAL,D,LP_LX11_CINIT ;Controller init
0071          DPT_STORE CRB,CRB$L_INTD+VEC$L_UNITINIT,D,LP_LX11_INIT ;Unit init
0076          DPT_STORE DDB,DDB$L_DDT,D,LP$DDT ;DDT address
007B          DPT_STORE END                     ;
0000
0000    ;
0000    ; DRIVER DISPATCH TABLE
0000    ;
0000
0000          DDTAB   LP,-                      ;DRIVER DISPATCH TABLE
0000                  STARTIO,-                 ;Start I/O operation
0000                  0,-                       ;Unsolicited interrupt
0000                  FUNCTABLE,-               ;Function table
0000                  +IOC$CANCELIO,-           ;Cancel I/O
0000                  0,-                       ;Register dump routine
0000                  0,-                       ;Size of diagnostic buffer
0000                  0                         ;Size of error log buffer
```

```
0038           .SBTTL  LP11/LS11/LV11 Function decision table
0038      ;
0038      ; LP11/LS11/LV11 FUNCTION DECISION TABLE
0038      ;
0038
0038      FUNCTABLE:                                  ;FUNCTION DECISION TABLE
0038               FUNCTAB ,-                         ;Legal functions
0038                       <SENSECHAR,-               ;Sense characteristics
0038                        SETCHAR,-                 ;Set characteristics
0038                        SENSEMODE,-               ;Sense mode
0038                        SETMODE,-                 ;Set mode
0038                        WRITELBLK,-               ;Write logical block
0038                        WRITEPBLK,-               ;Write physical block
0038                        WRITEVBLK>                ;Write virtual block
0040               FUNCTAB ,-                         ;LEGAL FUNCTIONS
0040                       <SENSECHAR,-               ;Sense characteristics
0040                        SETCHAR,-                 ;Set characteristics
0040                        SENSEMODE,-               ;Sense mode
0040                        SETMODE,-                 ;Set mode
0040                        WRITELBLK,-               ;Write logical block
0040                        WRITEPBLK,-               ;Write physical block
0040                        WRITEVBLK>                ;Write virtual block
0048               FUNCTAB LP_WRITE,<WRITELBLK,WRITEPBLK,WRITEVBLK> ;Write functions
0054               FUNCTAB LP_SETMODE,<SETCHAR,SETMODE> ;Set characteristics functions
0060               FUNCTAB +EXE$SENSEMODE,-           ;
0060                       <SENSECHAR,-               ;Sense characteristics
0060                        SENSEMODE>                ;Sense mode
```

```
                     006C          .SBTTL  Set characteristics and set mode function processing
                     006C     ;+
                     006C     ; LP_SETMODE - SET CHARACTERISTICS AND SET MODE FUNCTION PROCESSING
                     006C     ;
                     006C     ; THIS ROUTINE IS CALLED FROM THE FUNCTION DECISION TABLE DISPATCHER TO PROCESS
                     006C     ; A SET MODE FUNCTION TO A LINE PRINTER.
                     006C     ;
                     006C     ; INPUTS:
                     006C     ;
                     006C     ;       R0 = SCRATCH.
                     006C     ;       R1 = SCRATCH.
                     006C     ;       R2 = SCRATCH.
                     006C     ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                     006C     ;       R4 = CURRENT PROCESS PCB ADDRESS.
                     006C     ;       R5 = ASSIGNED DEVICE UCB ADDRESS.
                     006C     ;       R6 = ADDRESS OF CCB.
                     006C     ;       R7 = I/O FUNCTION CODE.
                     006C     ;       R8 = FUNCTION DECISION TABLE DISPATCH ADDRESS.
                     006C     ;       R9 = SCRATCH.
                     006C     ;       R10 = SCRATCH.
                     006C     ;       R11 = SCRATCH.
                     006C     ;       AP = ADDRESS OF FIRST FUNCTION DEPENDENT PARAMETER.
                     006C     ;
                     006C     ; OUTPUTS:
                     006C     ;
                     006C     ;       THE SPECIFIED CHARACTERISTICS ARE MOVED INTO THE DEVICE UCB AND THE
                     006C     ;       I/O IS COMPLETED.
                     006C     ;-
                     006C
                     006C     LP_SETMODE:                                     ;Set mode function processing
       51    6C   D0 006C             MOVL    P1(AP),R1                       ;Get address of characteristics
                     006F             IFNORD  #8,(R1),20$                     ;Can characteristics quadword be read?
             53   DD 0075             PUSHL   R3                              ;Save packet address
  50   0090 C5    9E 0077             MOVAB   UCB$L_LP_MUTEX(R5),R0           ;Get address of UCB mutex
    00000000'GF   16 007C             JSB     G^SCH$LOCKW                     ;Lock UCB for write access
       57    23   D1 0082             CMPL    #IO$_SETMODE,R7                 ;Set mode function?
             04   13 0085             BEQL    10$                             ;If EQL yes
    40 A5    61   B0 0087             MOVW    (R1),UCB$B_DEVCLASS(R5)         ;Set device class and type
 42 A5   02 A1   B0 008B     10$:    MOVW    2(R1),UCB$W_DEVBUFSIZ(R5)       ;Set default buffer size
 44 A5   04 A1   D0 0090             MOVL    4(R1),UCB$L_DEVDEPEND(R5)       ;Set device characteristics
    00000000'GF   16 0095             JSB     G^SCH$UNLOCK                    ;Unlock UCB
             53 8ED0 009B             POPL    R3                              ;Restore packet
       50    01   3C 009E             MOVZWL  #SS$_NORMAL,R0                  ;Set normal completion status
    00000000'GF   17 00A1             JMP     G^EXE$FINISHIOC                 ;
       50    0C   3C 00A7     20$:    MOVZWL  #SS$_ACCVIO,R0                  ;Set access violation status
    00000000'GF   17 00AA             JMP     G^EXE$ABORTIO                   ;
```

```
                         00B0        .SBTTL  Write function processing
                         00B0  ;+
                         00B0  ; LP_WRITE - WRITE FUNCTION PROCESSING
                         00B0  ;
                         00B0  ; THIS ROUTINE IS CALLED FROM THE FUNCTION DECISION TABLE DISPATCHER TO PROCESS
                         00B0  ; A WRITE PHYSICAL, WRITE LOGICAL, OR WRITE VIRTUAL FUNCTION TO A LINE PRINTER.
                         00B0  ;
                         00B0  ; INPUTS:
                         00B0  ;
                         00B0  ;       R0 = SCRATCH.
                         00B0  ;       R1 = SCRATCH.
                         00B0  ;       R2 = SCRATCH.
                         00B0  ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                         00B0  ;       R4 = CURRENT PROCESS PCB ADDRESS.
                         00B0  ;       R5 = ASSIGNED DEVICE UCB ADDRESS.
                         00B0  ;       R6 = ADDRESS OF CCB.
                         00B0  ;       R7 = I/O FUNCTION CODE.
                         00B0  ;       R8 = FUNCTION DECISION TABLE DISPATCH ADDRESS.
                         00B0  ;       R9 = SCRATCH.
                         00B0  ;       R10 = SCRATCH.
                         00B0  ;       R11 = SCRATCH.
                         00B0  ;       AP = ADDRESS OF FIRST FUNCTION DEPENDENT PARAMETER.
                         00B0  ;
                         00B0  ; OUTPUTS:
                         00B0  ;
                         00B0  ;       THE FUNCTION PARAMETERS ARE CHECKED AND THE USER'S BUFFER IS FORMATTED
                         00B0  ;       AND COPIED INTO A SYSTEM BUFFER FOR PROCESSING BY THE LINE PRINTER
                         00B0  ;       DRIVER.
                         00B0  ;-
                         00B0
                         00B0  LP_WRITE:                                      ;WRITE FUNCTION PROCESSING
             5B     D4   00B0          CLRL    R11                            ;Clear total number of overhead bytes
             5A     D4   00B2          CLRL    R10                            ;Assume write pass all function
         5E  5D     D0   00B4  FORMAT: MOVL    FP,SP                          ;Remove all temporaries from stack
       10F8  8F     BB   00B7          PUSHR   #^M<R3,R4,R5,R6,R7,AP>         ;Save registers
         58  6C     D0   00BB          MOVL    P1(AP),R8                      ;Get starting address of user buffer
      59  04 AC     3C   00BE          MOVZWL  P2(AP),R9                      ;Get length of user buffer
   03 44 A5  08     E1   00C2          BBC     #LP$V_PASSALL,UCB$L_DEVDEPEND(R5),5$ ; If CLR, not passall
         57  0B     D0   00C7          MOVL    #IO$_WRITEPBLK,R7              ;Force write physical
         57  0B     D1   00CA  5$:     CMPL    #IO$_WRITEPBLK,R7              ;Write physical block?
             1B     13   00CD          BEQL    10$                            ;If EQL yes
   3C A3  0C AC     D0   00CF          MOVL    P4(AP),IRP$B_CARCON(R3)        ;Insert carriage control information
    00000000'GF     16   00D4          JSB     G^EXE$CARRIAGE                 ;Translate carriage control information
     50   3C A3     9A   00DA          MOVZBL  IRP$B_CARCON(R3),R0            ;Get number of prefix control bytes
     51   3E A3     9A   00DE          MOVZBL  IRP$B_CARCON+2(R3),R1          ;Get number of suffix control bytes
         51  50     C0   00E2          ADDL    R0,R1                          ;Calculate number of carriage control bytes
  5A   20 A14B      9E   00E5          MOVAB   32(R1)[R11],R10                ;Calculate total number of overhead bytes
             59     D5   00EA  10$:    TSTL    R9                             ;Any buffer specified?
             09     13   00EC          BEQL    20$                            ;If EQL no
         50  58     7D   00EE          MOVQ    R8,R0                          ;Retrieve buffer parameters
    00000000'GF     16   00F1          JSB     G^EXE$WRITECHK                 ;Check accessibility of user buffer
  51   0C A94A      9E   00F7  20$:    MOVAB   12(R9)[R10],R1                 ;Calculate length of buffer required
    00000000'GF     16   00FC          JSB     G^EXE$BUFFRQUOTA               ;Check if process has sufficient quota
         03  50     E8   0102          BLBS    R0,25$                         ;If LBS quota ok
           0081     31   0105          BRW     45$                            ;If LBC quota check failure
    00000000'GF     16   0108  25$:    JSB     G^EXE$ALLOCBUF                 ;Allocate buffer for line printer output
         78  50     E9   010E          BLBC    R0,45$                         ;If LBC allocation failure
```

```
            53   6E   D0  0111          MOVL    (SP),R3                    ;Retrieve address of I/O packet
         2C A3   52   D0  0114          MOVL    R2,IRP$L_SVAPTE(R3)        ;Save address of buffered I/O packet
      50  0080   C4   D0  0118          MOVL    PCB$L_JIB(R4),R0           ;Get JIB address
         20 A0   51   C2  011D          SUBL    R1,JIB$L_BYTCNT(R0)        ;Adjust buffered I/O quota
         30 A3   51   B0  0121          MOVW    R1,IRP$W_BOFF(R3)          ;Set number of bytes charged to quota
              38 A3   D4  0125          CLRL    IRP$L_MEDIA(R3)            ;Clear line feed count in packet
         32 A3   59   B0  0128          MOVW    R9,IRP$W_BCNT(R3)          ;Insert size of user buffer
       52   0C   A2   9E  012C          MOVAB   12(R2),R2                  ;Get address of buffer data area
      50  0090   C5   9E  0130          MOVAB   UCB$L_LP_MUTEX(R5),R0      ;Get address of UCB mutex
      00000000'GF     16  0135          JSB     G^SCH$LOCKW                ;Lock UCB for write access
            57   08   D1  013B          CMPL    #IO$_WRITEPBLK,R7          ;Write pass all?
                 53   13  013E          BEQL    50$                        ;If EQL yes
            51   0C   A2  0140          SUBW    #12,R1                     ;Calculate actual length of data area
      54  009C   C5   9A  0143          MOVZBL  UCB$B_LP_CURSOR(R5),R4     ;Get current horizontal carriage position
       56   68   A5   3C  0148          MOVZWL  UCB$W_DEVSTS(R5),R6        ;Get current carriage return pending flag
      57  009D   C5   9A  014C          MOVZBL  UCB$B_LP_LINCNT(R5),R7     ;Get current line on page
       5A   42   A5   3C  0151          MOVZWL  UCB$W_DEVBUFSIZ(R5),R10    ;Get width of printer carriage
            5C   20   D0  0155          MOVL    #^X20,AP                   ;Assume printer does not have lower case
      02 44 A5   07   E1  0158          BBC     #LP$V_LOWER,UCB$L_DEVDEPEND(R5),35$ ;If CLR, no lower case
                 5C   D4  015D          CLRL    AP                         ;Set for printer with lower case
                 54   10  015F  35$:    BSBB    70$                        ;Insert prefix carriage control
                 59   D7  0161  30$:    DECL    R9                         ;Any more bytes to transfer to system buffer
                 07   19  0163          BLSS    40$                        ;If LSS no
            50   88   9A  0165          MOVZBL  (R8)+,R0                   ;Get next byte from user buffer
                 73   10  0168          BSBB    WRITE_BYTE                 ;Write byte in system buffer
                 F5   11  016A          BRB     30$                        ;
                 53   10  016C  40$:    BSBB    80$                        ;Insert suffix carriage control in buffer
       52   2C   A3   C2  016E          SUBL    IRP$L_SVAPTE(R3),R2        ;Calculate length of output plus header
    3A A3   52   0C   A3  0172          SUBW3   #12,R2,IRP$L_MEDIA+2(R3)   ;Calculate actual length of output buffer
       009C C5   54   90  0177          MOVB    R4,UCB$B_LP_CURSOR(R5)     ;Save current horizontal carriage position
68 A5   01   00   56   F0  017C          INSV    R6,#V_CRPEND,#1,UCB$W_DEVSTS(R5) ;Save carriage return pending
       009D C5   57   90  0182          MOVB    R7,UCB$B_LP_LINCNT(R5)     ;Save current line on page
                 12   11  0187          BRB     60$                        ;
          10F8   8F   BA  0189  45$:    POPR    #^M<R3,R4,R5,R6,R7,AP>     ;Restore registers
      00000000'GF     17  018D          JMP     G^EXE$ABORTIO              ;
         3A A3   59   B0  0193  50$:    MOVW    R9,IRP$L_MEDIA+2(R3)       ;Insert number of bytes to print
       62   68   59   28  0197          MOVC    R9,(R8),(R2)               ;Move characters to system buffer
          10F8   8F   BA  019B  60$:    POPR    #^M<R3,R4,R5,R6,R7,AP>     ;Restore registers
                 53   DD  019F          PUSHL   R3                         ;Save address of I/O packet
      50  0090   C5   9E  01A1          MOVAB   UCB$L_LP_MUTEX(R5),R0      ;Get address of UCB mutex
      00000000'GF     16  01A6          JSB     G^SCH$UNLOCK               ;Unlock UCB
                 53 8ED0  01AC          POPL    R3                         ;Restore address of I/O packet
      00000000'GF     17  01AF          JMP     G^EXE$QIODRVPKT            ;Queue I/O packet to driver
                          01B5
                          01B5  ;
                          01B5  ; SUBROUTINE TO INSERT CARRIAGE CONTROL IN BUFFER
                          01B5  ;
                          01B5
       7E   3C   A3   9A  01B5  70$:    MOVZBL  IRP$B_CARCON(R3),-(SP)     ;Get number of characters to output
                 1F   13  01B9          BEQL    100$                       ;If EQL none
       50   3D   A3   9A  01BB          MOVZBL  IRP$B_CARCON+1(R3),R0      ;Get character to output
                 0A   11  01BF          BRB     85$                        ;
       7E   3E   A3   9A  01C1  80$:    MOVZBL  IRP$B_CARCON+2(R3),-(SP)   ;Get number of characters to output
                 13   13  01C5          BEQL    100$                       ;If EQL none
       50   3F   A3   9A  01C7          MOVZBL  IRP$B_CARCON+3(R3),R0      ;Get character to output
                 08   12  01CB  85$:    BNEQ    90$                        ;If NEQ character specified
            50   0D   9A  01CD          MOVZBL  #C_CR,R0                   ;Get carriage return
```

```
                                                                      H  8
LPDRIVER                        - LP11/LS11/LV11 LINE PRINTER DRIVER      15-SEP-1984 23:59:04  VAX/VMS Macro V04-00         Page  10
V04-000                         Write function processing                  5-SEP-1984 00:14:57  [DRIVER.SRC]LPDRIVER.MAR;1         (1)

                     0B   10  01D0    403          BSBB     WRITE_BYTE            ;Write byte in system buffer
                50   0A   9A  01D2    404          MOVZBL   #C_LF,R0              ;Get line feed
                     06   10  01D5    405 90$:     BSBB     WRITE_BYTE            ;Write byte in system buffer
                  FB 6E   F5  01D7    406          SOBGTR   (SP),90$              ;Any more left to insert?
                     8E   D5  01DA    407 100$:    TSTL     (SP)+                 ;Remove count from stack
                     05       01DC    408          RSB                            ;
```

```
                              01DD          .SBTTL  Write byte into system buffer
                              01DD  ;
                              01DD  ; SUBROUTINE TO FORMAT AND FILL SYSTEM BUFFER WITH LINE PRINTER OUTPUT ONE BYTE
                              01DD  ; AT A TIME.
                              01DD  ;
                              01DD
                              01DD  WRITE_BYTE:                                            ;WRITE BYTE INTO BUFFER
 49 00000441'EF    50    E0   01DD          BBS     R0,CONTROL_TAB,40$               ;If Set, Control character
          6E 56    00    E4   01E5          BBSC    #V_CRPEND,R6,60$                 ;If SET, carriage return pending
 03 00000461'EF    50    E1   01E9  5$:     BBC     R0,LOWERCASE_TAB,10$             ;If CLR, not lower case
             50    5C    C2   01F1          SUBL    AP,R0                            ;Convert character to upper case
                              01F4  10$:
             5A    54    D1   01F4          CMPL    R4,R10                           ;Still room on current line?
                   1A    1F   01F7          BLSSU   15$                              ;If LSS, yes
       15 44 A5    06    E1   01F9          BBC     #LP$V_TRUNCATE,UCB$L_DEVDEPEND(R5),15$ ;If CLEAR, notruncate
       1E 44 A5    04    E1   01FE          BBC     #LP$V_WRAP,UCB$L_DEVDEPEND(R5),30$ ; If CLR, then nowrap
                              0203
                   50    DD   0203  11$:    PUSHL   R0                               ;Save the current character
             50    0D    9A   0205          MOVZBL  #C_CR,R0                         ;Get carriage return code
                   D3    10   0208          BSBB    WRITE_BYTE                       ;Insert code in system buffer
             50    0A    9A   020A          MOVZBL  #C_LF,R0                         ;Set line feed character
                 009B    30   020D          BSBW    110$                             ;Insert line feed into system buffer
                   50  8ED0   0210          POPL    R0                               ;Restore current character
                              0213
                   54    D6   0213  15$:    INCL    R4                               ;Increment horizontal position
                   51    D7   0215  20$:    DECL    R1                               ;Any room left in system buffer?
                   12    19   0217          BLSS    37$                              ;If less than, no
       04 44 A5    09    E0   0219  25$:    BBS     #LP$V_FALLBACK, UCB$L_DEVDEPEND(R5), 35$      ;if set, fallback
             82    50    90   021E          MOVB    R0,(R2)+                         ;Insert character in system buffer
                         05   0221  30$:    RSB                                      ;
                              0222
 82   00000481'FF40      90   0222  35$:    MOVB    @TRANS_TAB[R0],(R2)+             ;move translated character into system buffe
                         05   022A          RSB                                      ;return to caller, for another byte
                              022B
                 0092    31   022B  37$:    BRW     150$                             ;no room in system buffer
                              022E
                              022E  ;
                              022E  ; CONTROL CHARACTER ENCOUNTERED
                              022E  ;
                              022E
          7F 8F    50    91   022E  40$:    CMPB    R0,#^X7F                         ;Delete Character?
                   05    12   0232          BNEQ    45$                              ;neg, not a delete character
       BB 44 A5    02    E0   0234          BBS     #LP$V_PRINTALL,UCB$L_DEVDEPEND(R5),10$ ; If SET, allow delete charac
                   E6    1E   0239  45$:    BGEQU   30$                              ;If GEQU, non-printable character(multi)
             50    0D    D1   023B          CMPL    #C_CR,R0                         ;Carriage return?
                   08    1F   023E          BLSSU   50$                              ;If LSS no
                   22    1A   0240          BGTRU   70$                              ;If GTRU no
       74 44 A5    00    E0   0242          BBS     #LP$V_CR,UCB$L_DEVDEPEND(R5),140$ ;If SET, carriage return required
             56    01    C8   0247          BISL    #M_CRPEND,R6                     ;Set carriage return pending
                         05   024A          RSB                                      ;
          08 56    00    E4   024B  50$:    BBSC    #V_CRPEND,R6,60$                 ;If SET, carriage return pending
       C1 44 A5    02    E0   024F          BBS     #LP$V_PRINTALL,UCB$L_DEVDEPEND(R5),20$ ;If SET, print character
                 FFCA    31   0254          BRW     30$                              ;Exit this is nonprintable
                   50    DD   0257  60$:    PUSHL   R0                               ;Save current character
             50    0D    9A   0259          MOVZBL  #C_CR,R0                         ;Get carriage return character
                   5D    10   025C          BSBB    140$                             ;Insert carriage return in buffer
                   50  8ED0   025E          POPL    R0                               ;Retrieve current character
          FF79   31  0261          BRW     WRITE_BYTE                    ;
                     0264
                     0264  ;
                     0264  ; CHARACTER IS A TAB, LINE FEED, VERTICLE TAB, OR FORM FEED
                     0264  ;
                     0264
       50   09   D1  0264  70$:    CMPL    #C_TAB,R0                     ;Tabulation character?
            E2   1A  0267          BGTRU   50$                           ;If GTRU no
            17   1F  0269          BLSSU   80$                           ;If LSSU no
                     026B
                     026B  ;
                     026B  ; CHARACTER IS A TAB
                     026B  ;
                     026B
    E8 56   00   E4  026B          BBSC    #V_CRPEND,R6,60$              ;If SET, carriage return pending
 A1 44 A5   05   E0  026F          BBS     #LP$V_TAB,UCB$L_DEVDEPEND(R5),20$ ;If SET, do not expand TAB
       08 A4     9F  0274          PUSHAB  8(R4)                         ;Calculate next tab position
       6E   07   CA  0277          BICL    #7,(SP)                       ;Clear excess bits
       6E   54   C2  027A          SUBL    R4,(SP)                       ;Calculate blank count
       50   20   9A  027D          MOVZBL  #^A/ /,R0                     ;Set space character
            20   11  0280          BRB     100$                          ;
                     0282
                     0282  ;
                     0282  ; CHARACTER IS A LINE FEED, VERTICAL TAB, OR FORM FEED
                     0282  ;
                     0282
       50   0B   D1  0282  80$:    CMPL    #C_VT,R0                      ;Vertical tab?
            C4   13  0285          BEQL    50$                           ;If EQL yes
            22   1A  0287          BGTRU   110$                          ;If GTRU line feed
                     0289
                     0289  ;
                     0289  ; CHARACTER IS A FORM FEED
                     0289  ;
                     0289
    50   47 A5   9A  0289          MOVZBL  UCB$L_DEVDEPEND+3(R5),R0 ;Get number of lines per page
    7E   50   57   C3  028D        SUBL3   R7,R0,-(SP)                   ;Calculate number of lines to end of page
 09 44 A5   01   E1  0291          BBC     #LP$V_MECHFORM,UCB$L_DEVDEPEND(R5),90$ ;If CLR, no mechanical feed
    38 A3   8E   C0  0296          ADDL    (SP)+,IRP$L_MEDIA(R3)         ;Update number of lines printed
       50   0C   9A  029A          MOVZBL  #C_FF,R0                      ;Set form feed character
            17   11  029D          BRB     120$                          ;
       50   0A   9A  029F  90$:    MOVZBL  #C_LF,R0                      ;Set line feed character
          FF38   30  02A2  100$:   BSBW    WRITE_BYTE                    ;Insert byte in system buffer
       FA   6E   F5  02A5          SOBGTR  (SP),100$                     ;Any more bytes to insert?
            8E   D5  02A8          TSTL    (SP)+                         ;Remove loop count from stack
                 05  02AA          RSB                                   ;
                     02AB
                     02AB  ;
                     02AB  ; CHARACTER IS A LINE FEED
                     02AB  ;
                     02AB
                     02AB  110$:
            57   D6  02AB          INCL    R7                            ;Increment line position on page
       38 A3     D6  02AD          INCL    IRP$L_MEDIA(R3)               ;Increment number of lines printed
    47 A5   57   91  02B0          CMPB    R7,UCB$L_DEVDEPEND+3(R5)      ;End of page?
            02   12  02B4          BNEQ    130$                          ;If NEQ no
            57   D4  02B6  120$:   CLRL    R7                            ;Clear line position on page
       56   01   CA  02B8  130$:   BICL    #M_CRPEND,R6                  ;Clear carriage return pending
```

```
              54   D4  02BB   140$:   CLRL    R4                        ;Clear horizontal position
            FF55   31  02BD           BRW     20$                       ;
                       02C0
                       02C0   ;
                       02C0   ; OUTPUT WILL NOT FIT IN ALLOCATED BUFFER
                       02C0   ;
                       02C0
         50   2C A3   D0  02C0   150$:   MOVL    IRP$L_SVAPTE(R3),R0       ;Get address of buffer to deallocate
              2C A3   D4  02C4           CLRL    IRP$L_SVAPTE(R3)          ;Indicate no buffer allocated
         5A   08 A0   3C  02C7           MOVZWL  IRP$W_SIZE(R0),R10        ;Save size of buffer
       00000000'GF    16  02CB           JSB     G^EXE$DEANONPAGED         ;Deallocate buffer
         5E   E8 AD   9E  02D1           MOVAB   -4*6(FP),SP               ;Remove all temporaries from stack
            10F8 8F   BA  02D5           POPR    #^M<R3,R4,R5,R6,R7,AP>    ;Restore registers
       50   0080 C4   D0  02D9           MOVL    PCB$L_JIB(R4),R0          ;Get JIB address
         20 A0   5A   C0  02DE           ADDL    R10,JIB$L_BYTCNT(R0)      ;Adjust byte count quota
              5B 20   C0  02E2           ADDL    #32,R11                   ;Adjust count of overhead bytes
                 53   DD  02E5           PUSHL   R3                        ;Save address of I/O packet
       50   0090 C5   9E  02E7           MOVAB   UCB$L_LP_MUTEX(R5),R0     ;Get address of UCB mutex
       00000000'GF    16  02EC           JSB     G^SCH$UNLOCK              ;Unlock UCB
              53 8ED0     02F2           POPL    R3                        ;Restore address of I/O packet
            FDBC   31     02F5           BRW     FORMAT                    ;Try again
                          02F8
```

```
                              02F8         .SBTTL  Line printer driver
                              02F8  ;+
                              02F8  ; STARTIO - START I/O OPERATION ON LINE PRINTERS
                              02F8  ;
                              02F8  ; THIS ROUTINE IS ENTERED WHEN THE ASSOCIATED UNIT IS IDLE AND A PACKET
                              02F8  ; IS AVAILABLE.
                              02F8  ;
                              02F8  ; INPUTS:
                              02F8  ;
                              02F8  ;       R3 = ADDRESS OF I/O REQUEST PACKET.
                              02F8  ;       R5 = UCB ADDRESS FOR IDLE UNIT.
                              02F8  ;
                              02F8  ; OUTPUTS:
                              02F8  ;
                              02F8  ;       NO EXPLICIT OUTPUTS - THE UNIT IS IN WAITING FOR INTERRPUT STATE
                              02F8  ;                               OR THE I/O IS COMPLETE.
                              02F8  ;-
                              02F8
                              02F8  STARTIO:
          53    58 A5   D0    02F8         MOVL    UCB$L_IRP(R5),R3           ;Retrieve address of I/O packet
                3A A3   B0    02FC         MOVW    IRP$L_MEDIA+2(R3),-
                7C A5         02FF                 UCB$W_BOFF(R5)             ;Set number of characters to print
          53    78 A5   D0    0301         MOVL    UCB$L_SVAPTE(R5),R3        ;Get address of system buffer
          53    0C A3   9E    0305         MOVAB   12(R3),R3                  ;Get address of data area
          54    24 A5   D0    0309         MOVL    UCB$L_CRB(R5),R4           ;Get address of CRB
          54    2C B4   D0    030D         MOVL    @CRB$L_INTD+VEC$L_IDB(R4),R4 ;Get device CSR address
                              0311  ;
                              0311  ; START NEXT OUTPUT SEQUENCE
                              0311  ;
                              0311
       50   54     02   C1    0311  10$:   ADDL3   #LP_DBR,R4,R0              ;Calculate address of data buffer register
          51     7C A5  3C    0315         MOVZWL  UCB$W_BOFF(R5),R1          ;Get number of characters remaining
       52   8080   8F   B0    0319         MOVW    #^X8080,R2                 ;Get control register test mask
                   16   11    031E         BRB     25$                        ;Start output
            64     52   B3    0320  20$:   BITW    R2,(R4)                    ;Printer ready or have paper problem?
                   17   15    0323         BLEQ    30$                        ;If LEQ not ready or paper problem
            60     83   90    0325         MOVB    (R3)+,(R0)                 ;Output next character
  7E    00000000'GF     01    78    0328   ASHL    #1, G^EXE$GL_UBDELAY,-(SP)       ;Delay 3*2 u-seconds
               FD 6E    F4    0330  24$:   SOBGEQ  (SP),24$                   ;Delay loop calibrated to machine speed
            5E     04   C0    0333         ADDL    #4,SP                      ;Pop extra longword off stack
               E7 51    F4    0336  25$:   SOBGEQ  R1,20$                     ;Any more characters to output?
              009D      31    0339         BRW     70$                        ;All done, BRW to set return status
                              033C
                              033C  ;
                              033C  ; PRINTER IS NOT READY OR HAS PAPER PROBLEM
                              033C  ;
                              033C
                   30   12    033C  30$:   BNEQ    40$                        ;If NEQ paper problem
    7C A5   51     01   A1    033E         ADDW3   #1,R1,UCB$W_BOFF(R5)       ;Save number of characters remaining
                              0343         DSBINT  UCB$B_DIPL(R5)             ;Disable interrupts
       64   0080   8F   B3    034A         BITW    #^X80,LP_CSR(R4)           ;Is it ready now?
                   16   12    034F         BNEQ    35$                        ;If NEQ, yes its ready
       64     40   8F   88    0351         BISB    #^X40,LP_CSR(R4)           ;Set interrupt enable
                              0355         WFIKPCH 50$,#12                    ;Wait for ready interrupt
                              035F         IOFORK                             ;Create a fork process
                   AA   11    0365         BRB     10$                        ;  ...and start next output
                              0367
```

```
                                  0367  35$:
                                  0367          ENBINT                             ;Enable system interrupts
                   64       B4    036A          CLRW    LP_CSR(R4)                 ;Disable device interrupts
                   A3       11    036C          BRB     10$                        ;Go transfer more characters
                                  036E  ;
                                  036E  ; PRINTER HAS PAPER PROBLEM
                                  036E  ;
                                  036E  ;
              0098 C5       D4    036E  40$:    CLRL    UCB$L_LP_OFLCNT(R5)        ;Clear offline counter
        7C A5     51  01    A1    0372          ADDW3   #1,R1,UCB$W_BOFF(R5)       ;Save number of characters remaining
                   64       B4    0377  50$:    CLRW    LP_CSR(R4)                 ;Disable printer interrupt
                                  0379          SETIPL  UCB$B_FIPL(R5)             ;Lower to fork level
                   64       B5    037D          TSTW    LP_CSR(R4)                 ;Printer still have paper problem?
                   08       19    037F          BLSS    55$                        ;If LSS yes
           0094 C5    0F    D0    0381          MOVL    #15,UCB$L_LP_TIMEOUT(R5)   ;Set timeout value
                 FF88       31    0386          BRW     10$                        ; ...and start next output
        53 64 A5    03      E0    0389  55$:    BBS     #UCB$V_CANCEL,UCB$W_STS(R5),80$ ;If SET, cancel I/O operation
                                  038E
         01   0094 C5       F1    038E          ACBL    UCB$L_LP_TIMEOUT(R5),#1,-
         0028 0098 C5             0393                  UCB$L_LP_OFLCNT(R5),60$ ;Skip until timeout
                                  0398
              0098 C5       D4    0398          CLRL    UCB$L_LP_OFLCNT(R5)        ;Reset counter
          00000780 8F       D1    039C          CMPL    #LP_HRCNT,-                ;One hour timeout?
              0094 C5             03A2                  UCB$L_LP_TIMEOUT(R5)
                   05       1B    03A5          BLEQU   57$                        ;If LSS yes and dont increment
           0094 C5    02    C4    03A7          MULL    #2,UCB$L_LP_TIMEOUT(R5)    ;Double message timeout value
                   18       BB    03AC  57$:    PUSHR   #^M<R3,R4>                 ;Save registers
              54   05       9A    03AE          MOVZBL  #MSG$_DEVOFFLIN,R4         ;Set up message type
      53    00000000'GF     9E    03B1          MOVAB   G^SYS$GL_OPRMBX,R3         ;Address target mailbox
            00000000'GF     16    03B8          JSB     G^EXE$SNDEVMSG             ;Send message ignore error
                   18       BA    03BE          POPR    #^M<R3,R4>                 ;Restore registers
                                  03C0  60$:    DSBINT  UCB$B_DIPL(R5)             ;Disable interrupts
                                  03C7          WFIKPCH 50$,#2                     ;Wait for a timeout
                                  03D1          IOFORK                             ;Create for process
                   9E       11    03D7          BRB     50$                        ;
                                  03D9
                                  03D9  ;
                                  03D9  ; I/O OPERATION SUCCESSFULLY COMPLETED
                                  03D9  ;
                                  03D9
              50   01       3C    03D9  70$:    MOVZWL  #SS$_NORMAL,R0             ;Set normal completion status
                 7C A5      B4    03DC          CLRW    UCB$W_BOFF(R5)             ;Correct remaining character count
                   03       11    03DF          BRB     90$                        ;
                                  03E1
                                  03E1  ;
                                  03E1  ; I/O OPERATION CANCELED
                                  03E1  ;
                                  03E1
              50   2C       3C    03E1  80$:    MOVZWL  #SS$_ABORT,R0              ;Set operation aborted status
          53     58 A5      D0    03E4  90$:    MOVL    UCB$L_IRP(R5),R3           ;Retrieve address of I/O packet
          51     38 A3      3C    03E8          MOVZWL  IRP$L_MEDIA(R3),R1         ;Get number of lines printed
       7E A5     7C A5      A2    03EC          SUBW    UCB$W_BOFF(R5),UCB$W_BCNT(R5) ;Calculate number of characters
 50    10   10   7E A5      F0    03F1          INSV    UCB$W_BCNT(R5),#16,#16,R0 ;Insert number of characters in status
                                  03F7          REQCOM                             ;Complete I/O request
```

```
                          03FD         .SBTTL  LP11/LS11/LV11 Line printer interrupt dispatcher
                          03FD   ;+
                          03FD   ; LP$INT - LP11/LS11/LV11 LINE PRINTER INTERRUPT DISPATCHER.
                          03FD   ;
                          03FD   ; THIS ROUTINE IS ENTERED VIA A JSB INSTRUCTION WHEN AN INTERRUPT OCCURS ON AN
                          03FD   ; LP11/LS11/LV11 LINE PRINTER CONTROLLER. THE STATE OF THE STACK ON ENTRY IS:
                          03FD   ;
                          03FD   ;       00(SP) = ADDRESS OF IDB ADDRESS.
                          03FD   ;       04(SP) = SAVED R3.
                          03FD   ;       08(SP) = SAVED R4.
                          03FD   ;       12(SP) = SAVED R5.
                          03FD   ;       16(SP) = INTERRUPT PC.
                          03FD   ;       20(SP) = INTERRUPT PSL.
                          03FD   ;
                          03FD   ; INTERRUPT DISPATCHING OCCURS AS FOLLOWS:
                          03FD   ;
                          03FD   ;       IF THE INTERRUPT IS EXPECTED, THEN THE DRIVER IS CALLED AT ITS INTERRUPT
                          03FD   ;       WAIT ADDRESS. ELSE THE INTERRUPT IS DISMISSED.
                          03FD   ;-
                          03FD
                          03FD   LP$INT::                                                :Entry from dispatch
           53    9E   D0  03FD         MOVL    @(SP)+,R3                               ;Get address of IDB
           54    63   7D  0400         MOVQ    IDB$L_CSR(R3),R4                        ;Get controller CSR and owner UCB address
     09 64 A5    01   E5  0403         BBCC    #UCB$V_INT,UCB$W_STS(R5),10$ ;If CLR, interrupt not expected
                 64   B4  0408         CLRW    (R4)                                    ;Disable output interrupts
        53    10 A5   D0  040A         MOVL    UCB$L_FR3(R5),R3                        ;Restore remainder of driver context
              0C B5   16  040E         JSB     @UCB$L_FPC(R5)                          ;Call driver at interrupt wait address
           50    8E   7D  0411   10$:  MOVQ    (SP)+,R0                                ;Restore registers
           52    8E   7D  0414         MOVQ    (SP)+,R2                                ;
           54    8E   7D  0417         MOVQ    (SP)+,R4                                ;
                      02  041A         REI                                             ;
```

```
                                   041B          .SBTTL  Line printer unit initialization
                                   041B  ;+
                                   041B  ; LP_LX11_INIT - LINE PRINTER UNIT INITIALIZATION
                                   041B  ;
                                   041B  ; THIS ROUTINE IS CALLED AT SYSTEM STARTUP AND AFTER A POWER FAILURE. THE
                                   041B  ; ONLINE BIT IS SET FOR THE SPECIFIED UNIT.
                                   041B  ;
                                   041B  ; INPUTS:
                                   041B  ;
                                   041B  ;       R5 = ADDRESS OF DEVICE UCB.
                                   041B  ;
                                   041B  ; OUTPUTS:
                                   041B  ;
                                   041B  ;       THE ONLINE BIT IS SET IN THE DEVICE UCB AND THE ADDRESS OF THE UCB
                                   041B  ;       IS FILLED INTO THE IDB OWNER FIELD.
                                   041B  ;-
                                   041B
                                   041B  LP_LX11_INIT:                                   ;LINE PRINTER UNIT INITIALIZATION
                 64 A5    10   A8  041B          BISW    #UCB$M_ONLINE,UCB$W_STS(R5) ;Set unit online
                 50    24 A5   D0  041F          MOVL    UCB$L_CRB(R5),R0            ;Get address of CRB
                 50    2C A0   D0  0423          MOVL    CRB$L_INTD+VEC$L_IDB(R0),R0 ;Get address of IDB
                 04 A0    55   D0  0427          MOVL    R5,IDB$L_OWNER(R0)          ;Set address of device UCB
                               05  042B          RSB                                 ;Return
                                   042C
                                   042C  LP_LX11_CINIT:                              ;CONTROLLER INITIALIZATION
                 50    18 A5   D0  042C          MOVL    IDB$L_UCBLST(R5),R0         ;Get address of UCB
               0094 C0    0F   D0  0430          MOVL    #15,UCB$L_LP_TIMEOUT(R0);Set timeout value
00000481'EF  00000485'EF       9E  0435          MOVAB   FALLTAB,TRANS_TAB           ;Get address of fallback table
                               05  0440          RSB                                 ;
                                   0441
```

```
                0441          .SBTTL  Tables for lowercase and control characters
                0441    ;
                0441    ;       Bit table to distinguish control characters
                0441    ;
                0441    CONTROL_TAB:
            FFFF 0441           .WORD ^B1111111111111111
            FFFF 0443           .WORD ^B1111111111111111
        00000000 0445           .LONG 0
        00000000 0449           .LONG 0
            0000 044D           .WORD 0
            8000 044F           .WORD ^B1000000000000000
            FFFF 0451           .WORD ^B1111111111111111
            FFFF 0453           .WORD ^B1111111111111111
        00000000 0455           .LONG 0
        00000000 0459           .LONG 0
        00000000 045D           .LONG 0
                0461
                0461    ;
                0461    ;       Bit table to distinguish lower case characters
                0461    ;
                0461    LOWERCASE_TAB:
        00000000 0461           .LONG 0
        00000000 0465           .LONG 0
        00000000 0469           .LONG 0
            FFFE 046D           .WORD ^B1111111111111110
            07FF 046F           .WORD ^B0000011111111111
        00000000 0471           .LONG 0
        00000000 0475           .LONG 0
        00000000 0479           .LONG 0
            FFFF 047D           .WORD ^B1111111111111111
            3FFE 047F           .WORD ^B0011111111111110
                0481
                0481    ;
                0481    ;       Pointer to the fallback tables
                0481    ;
                0481    TRANS_TAB:
       00000485' 0481           .LONG       FALLTAB
                0485
                0485
```

```
0485             .SBTTL  FALLBACK - table that will create fallback presentation
0485    ;++
0485    ;FALLBACK - TABLE TO ALLOW THE TERMINAL TO DO FALLBACK PRESENTATION OF
0485    ;               8BIT CHARACTERS on 7 bit terminals
0485    ;
0485    ; Description:
0485    ;       The following macros generate 1 table.  The table is a 256 byte
0485    ; table with the single character fallback representation of all the
0485    ; characters that can be represented by a single character, those with
0485    ; no fallback presentation at all are represented by the _ character,
0485    ;
0485    ;--
0485             .macro  $fallini
0485    $$=0
0485    .repeat 256
0485    .IF LE $$-<^X9F>        ; EVERYTHING BUT THE MULTINATIONAL SET SHOULD
0485                            ; ECHO AS ITSELF.
0485             .byte $$
0485    .IFF
0485             .BYTE ^A/_/
0485    .ENDC
0485    $$=$$+1
0485    .endr
0485    $$$=.
0485             .endm   $fallini
```

```
0485  ;++
0485  ; $FALL - generates the table entry for a given character
0485  ;
0485  ; Inputs:
0485  ;
0485  ;         CHARH - COLUMN IN THE ASCII TABLE.
0485  ;         CHARL - ROW IN THE ASCII TABLE.
0485  ;         FALLBACK - String that is the fallback representation
0485  ;         COUNT - Number of times to repeat this character
0485  ;--
0485              .MACRO  $FALL   CHARH,CHARL,FALLBACK,COUNT=1
0485  .=FALLTAB+<CHARH*16>+CHARL
0485  .REPEAT COUNT
0485  .NCHR   SLEN,^\FALLBACK\
0485  .IF EQ  SLEN-1
0485              .BYTE   ^A/FALLBACK/
0485  .ENDR
0485              .ENDM   $FALL
```

```
0485   807 ;++
0485   808 ; $FALLEND - GENERATES END CONDITIONS FOR THE FALLBACK TABLE
0485   809 ;
0485   810 ; Description:
0485   811 ;
0485   812 ;       Resets the . to the end of the fallback table
0485   813 ;
0485   814 ; Inputs:
0485   815 ;
0485   816 ;       None
0485   817 ;--
0485   818         .MACRO  $FALLEND
0485   819 .=$$$
0485   820         .ENDM   $FALLEND
```

```
0485
0485  FALLTAB::
0485          $FALLINI
0585          $FALL     10,1,!
0527          $FALL     10,2,c
0528          $FALL     10,3,L
0529          $FALL     10,5,Y
052B          $FALL     10,8,0
052E          $FALL     10,10,a
0530          $FALL     11,0,o
0536          $FALL     11,1,+
0537          $FALL     11,2,2
0538          $FALL     11,3,3
0539          $FALL     11,5,u
053B          $FALL     11,7,.
053D          $FALL     11,9,1
053F          $FALL     11,10,o
0540          $FALL     11,15,?
0545          $FALL     12,0,A,6
054B          $FALL     12,7,C
054D          $FALL     12,8,E,4
0551          $FALL     12,12,I,4
0555          $FALL     13,1,N
0557          $FALL     13,2,O,5
055C          $FALL     13,8,O
055E          $FALL     13,9,U,4
0562          $FALL     13,13,Y
0563          $FALL     14,0,a,6
056B          $FALL     14,7,c
056D          $FALL     14,8,e,4
0571          $FALL     14,12,i,4
0575          $FALL     15,1,n
0577          $FALL     15,2,o,5
057C          $FALL     15,8,o
057E          $FALL     15,9,u,4
0582          $FALL     15,13,y
0583          $FALLEND
0585
0585
0585  LP_END:                                         ;Address of last location in driver
0585
0585          .END
```

```
$$                              = 00000100                 IOC$RETURN                         ********    X   03
$$$                             = 00000585 R       03      IOC$WFIKPCH                        ********    X   03
$$OP                            = 00000002                 IRP$B_CARCON                     = 0000003C
AT$_UBA                         = 00000001                 IRP$L_MEDIA                      = 00000038
CONTROL_TAB                       00000441 R       03      IRP$L_SVAPTE                     = 0000002C
CRB$L_INTD                      = 00000024                 IRP$W_BCNT                       = 00000032
C_CR                            = 0000000D                 IRP$W_BOFF                       = 00000030
C_FF                            = 0000000C                 IRP$W_SIZE                       = 00000008
C_LF                            = 0000000A                 JIB$L_BYTCNT                     = 00000020
C_TAB                           = 00000009                 LOWERCASE_TAB                      00000461 R      03
C_VT                            = 0000000B                 LP$DDT                             00000000 RG     03
DC$_LP                          = 00000043                 LP$INT                             000003FD RG     03
DDB$L_DDT                       = 0000000C                 LP$M_MECHFORM                    = 00000002
DEV$M_AVL                         ********    X    02      LP$M_TRUNCATE                    = 00000040
DEV$M_CCL                         ********    X    02      LP$V_CR                          = 00000000
DEV$M_NNM                         ********    X    02      LP$V_FALLBACK                    = 00000009
DEV$M_ODV                         ********    X    02      LP$V_LOWER                       = 00000007
DEV$M_REC                         ********    X    02      LP$V_MECHFORM                    = 00000001
DPT$C_LENGTH                    = 00000038                 LP$V_PASSALL                     = 00000008
DPT$C_VERSION                   = 00000004                 LP$V_PRINTALL                    = 00000002
DPT$INITAB                        00000038 R       02      LP$V_TAB                         = 00000005
DPT$REINITAB                      00000067 R       02      LP$V_TRUNCATE                    = 00000006
DPT$TAB                           00000000 R       02      LP$V_WRAP                        = 00000004
DYN$C_CRB                       = 00000005                 LP$_LP11                         = 00000001
DYN$C_DDB                       = 00000006                 LP_CSR                             00000000
DYN$C_DPT                       = 0000001E                 LP_DBR                             00000002
DYN$C_UCB                       = 00000010                 LP_END                             00000585 R      03
EXE$ABORTIO                       ********    X    03      LP_HRCNT                         = 00000780
EXE$ALLOCBUF                      ********    X    03      LP_LX11_CINIT                      0000042C R      03
EXE$BUFFRQUOTA                    ********    X    03      LP_LX11_INIT                       0000041B R      03
EXE$CARRIAGE                      ********    X    03      LP_SETMODE                         0000006C R      03
EXE$DEANONPAGED                   ********    X    03      LP_WRITE                           000000B0 R      03
EXE$FINISHIOC                     ********    X    03      MASKH                            = 00000080
EXE$GL_UBDELAY                    ********    X    03      MASKL                            = 08000000
EXE$IOFORK                        ********    X    03      MSG$_DEVOFFLIN                   = 00000005
EXE$QIODRVPKT                     ********    X    03      M_CRPEND                         = 00000001
EXE$SENSEMODE                     ********    X    03      P1                               = 00000000
EXE$SNDEVMSG                      ********    X    03      P2                               = 00000004
EXE$WRITECHK                      ********    X    03      P3                               = 00000008
FALLTAB                           00000485 RG      03      P4                               = 0000000C
FORMAT                            000000B4 R       03      P5                               = 00000010
FUNCTABLE                         00000038 R       03      P6                               = 00000014
FUNCTAB_LEN                     = 00000034                 PCB$L_JIB                        = 00000080
IDB$L_CSR                       = 00000000                 PR$_IPL                          = 00000012
IDB$L_OWNER                     = 00000004                 SCH$LOCKW                          ********    X   03
IDB$L_UCBLST                    = 00000018                 SCH$UNLOCK                         ********    X   03
IO$_SENSECHAR                   = 0000001B                 SIZ...                           = 00000001
IO$_SENSEMODE                   = 00000027                 SLEN                             = 00000001
IO$_SETCHAR                     = 0000001A                 SS$_ABORT                        = 0000002C
IO$_SETMODE                     = 00000023                 SS$_ACCVIO                       = 0000000C
IO$_VIRTUAL                     = 0000003F                 SS$_NORMAL                       = 00000001
IO$_WRITELBLK                   = 00000020                 STARTIO                            000002F8 R      03
IO$_WRITEPBLK                   = 0000000B                 SYS$GL_OPRMBX                      ********    X   03
IO$_WRITEVBLK                   = 00000030                 TRANS_TAB                          00000481 R      03
IOC$CANCELIO                      ********    X    03      UCB$B_DEVCLASS                   = 00000040
IOC$MNTVER                        ********    X    03      UCB$B_DEVTYPE                    = 00000041
IOC$REQCOM                        ********    X    03      UCB$B_DIPL                       = 0000005E
```

```
                                            I 9
LPDRIVER                   - LP11/LS11/LV11 LINE PRINTER DRIVER    15-SEP-1984 23:59:04  VAX/VMS Macro V04-00        Page  24
Symbol table                                                        5-SEP-1984 00:14:57  [DRIVER.SRC]LPDRIVER.MAR;1        (6)

UCB$B_FIPL                 = 0000000B
UCB$B_LP_CURSOR              0000009C
UCB$B_LP_LINCNT              0000009D
UCB$B_SPARE                  0000009E
UCB$K_LENGTH               = 00000090
UCB$K_SIZE                 = 000000A0
UCB$L_CRB                  = 00000024
UCB$L_DEVCHAR              = 00000038
UCB$L_DEVCHAR2             = 0000003C
UCB$L_DEVDEPEND            = 00000044
UCB$L_FPC                  = 0000000C
UCB$L_FR3                  = 00000010
UCB$L_IRP                  = 00000058
UCB$L_LP_MUTEX               00000090
UCB$L_LP_OFLCNT              00000098
UCB$L_LP_TIMEOUT             00000094
UCB$L_SVAPTE               = 00000078
UCB$M_ONLINE               = 00000010
UCB$V_CANCEL               = 00000003
UCB$V_INT                  = 00000001
UCB$W_BCNT                 = 0000007E
UCB$W_BOFF                 = 0000007C
UCB$W_DEVBUFSIZ            = 00000042
UCB$W_DEVSTS               = 00000068
UCB$W_STS                  = 00000064
VEC$L_IDB                  = 00000008
VEC$L_INITIAL              = 0000000C
VEC$L_UNITINIT             = 00000018
V_CRPEND                   = 00000000
WRITE_BYTE                   000001DD R     03

                                    +----------------+
                                    ! Psect synopsis !
                                    +----------------+

PSECT name                   Allocation          PSECT No.  Attributes
----------                   ----------          ---------  ----------
.  ABS  .                    00000000  (    0.)  00 (  0.)  NOPIC    USR    CON    ABS    LCL NOSHR NOEXE NORD   NOWRT NOVEC BYTE
$ABS$                        000000A0  (  160.)  01 (  1.)  NOPIC    USR    CON    ABS    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$105_PROLOGUE              0000007C  (  124.)  02 (  2.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC BYTE
$$$115_DRIVER                00000585  ( 1413.)  03 (  3.)  NOPIC    USR    CON    REL    LCL NOSHR   EXE   RD     WRT NOVEC LONG

                                    +------------------------+
                                    ! Performance indicators !
                                    +------------------------+

Phase                    Page faults    CPU Time       Elapsed Time
-----                    -----------    --------       ------------
Initialization                   29    00:00:00.04    00:00:00.80
Command processing              105    00:00:00.38    00:00:04.34
Pass 1                          565    00:00:17.62    00:01:02.00
Symbol table sort                 0    00:00:02.34    00:00:09.09
Pass 2                          167    00:00:03.52    00:00:11.18
Symbol table output              18    00:00:00.11    00:00:00.70
Psect synopsis output             2    00:00:00.01    00:00:00.01
Cross-reference output            0    00:00:00.00    00:00:00.00
Assembler run totals            888    00:00:24.03    00:01:28.13
```

```
The working set limit was 1950 pages.
149698 bytes (293 pages) of virtual memory were used to buffer the intermediate code.
There were 120 pages of symbol table space allocated to hold 2159 non-local and 54 local symbols.
863 source lines were read in Pass 1, producing 19 object records in Pass 2.
41 pages of virtual memory were used to define 38 macros.

                                              +----------------------------+
                                              ! Macro library statistics !
                                              +----------------------------+

Macro library name                             Macros defined
------------------                             --------------
_$255$DUA28:[SYS.OBJ]LIB.MLB;1                          22
_$255$DUA28:[SYSLIB]STARLET.MLB;2                       11
TOTALS (all libraries)                                  33

2424 GETS were required to define 33 macros.

There were no errors, warnings or information messages.

MACRO/LIS=LIS$:LPDRIVER/OBJ=OBJ$:LPDRIVER MSRC$:LPDRIVER/UPDATE=(ENH$:LPDRIVER)+EXECML$/LIB
```

0112 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

LCDRIVER
LIS

LPDRIVER
LIS

NODRIVER
LIS

MBXDRIVER
LIS